**Fraud Detection Using Random Forest**

A white circle with blue letters on it

Description automatically generated

A

ADM Course Project Report

in partial fulfilment of the degree

**Bachelor of Technology**

in

**Computer Science & Engineering**

**By**

V.Anushka 2303A51176

Aatiqah Harmine 2303A51860

M.Hanishka 2303A51684

Neha Fariyal 2303A51598

N.Akshitha 2303A51360

Under the guidance of

**Bediga Sharan**

**Assistant Professor**

**Submitted to**

**School of Computer Science and Artificial Intelligence**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**CERTIFICATE**

This is to certify that the **Applications of Data Mining – Course Project** Report entitled **“Online gaming transaction fraud detection”** is a record of bonafide work carried out by the student(s) **V.Anushka, Aatiqah Harmine, M.Hanishka, NehaFariyal, N.Akshitha** bearing Hallticket No(s) **2303A51176, 2303A51860, 2303A51684, 2303A51598, 2303A51360** during the academic year 2024-2025 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Warangal.

**Supervisor Head of the Department**

(Mr. Bediga Sharan) (Dr. M. Sheshikala)

Assistant Professor Professor

**Table of Contents**

|  |  |
| --- | --- |
| **Topic** | **Page No** |
| **Abstract** | **4** |
| **Objective of the project** | **5** |
| **Definitions of the elements used in project** | **6 - 7** |
| **Design** | **8 - 9** |
| **Screens** | **10** |
| **Implementation** | **11** |
| **Code** | **11 - 12** |
| **Result Screens** | **13 - 17** |
| **Conclusion** | **18** |

**Abstract**

The project "Online Gaming Transaction Fraud Detection" effectively addresses the growing concern of financial fraud in virtual gaming environments. It combines traditional rule-based detection methods with a machine learning approach, primarily utilizing a Random Forest classifier. The methodology starts with a detailed exploratory data analysis (EDA) to identify transaction patterns, anomalies, and high-risk behaviors. Based on these insights, the team defined a set of rules to flag potentially suspicious transactions—such as unusually high amounts or activities during odd hours—and subsequently trained the model on historical data labeled with these criteria. This hybrid approach enhances detection accuracy by ensuring that both clear-cut rules and subtle patterns are considered.

In the implementation, data preprocessing techniques like label encoding and feature scaling were applied to prepare the dataset. An engineered feature, amount\_to\_balance, adds further predictive value by reflecting the proportion of the transaction to the remaining balance. The model's performance was evaluated using accuracy scores, confusion matrices, and classification reports. Visual tools provided insight into class distributions, transaction behavior over time, and category-based fraud trends. Overall, the solution achieved strong performance and lays a foundation for scaling into real-time applications, with future enhancements potentially incorporating deep learning and adaptive behavior analysis to respond dynamically to evolving fraud tactics.

**OBJECTIVE OF THE PROJECT**

The objective of this project is to develop a robust and intelligent fraud detection system tailored specifically for online gaming transactions. As digital gaming platforms continue to grow in popularity, so does the volume and complexity of financial transactions made through in-game purchases and virtual currencies. With this growth comes an increasing risk of fraudulent activities, including unauthorized transactions and exploitation of digital wallets. This project aims to address these risks by identifying suspicious behaviors and transaction patterns using data-driven approaches.

To achieve this, the project integrates both rule-based filtering techniques and machine learning models. Rule-based logic helps flag transactions that exceed certain thresholds (e.g., unusually high or low amounts, transactions made during odd hours, or usage of certain wallets), while machine learning models are trained to detect more subtle fraud patterns based on historical data. Extensive exploratory data analysis (EDA) is conducted to uncover trends, correlations, and anomalies in the transaction dataset. Additionally, feature engineering is employed to derive new variables like the amount\_to\_balance ratio, further enhancing model performance.

A Random Forest Classifier is used due to its ability to handle complex, high-dimensional data and deliver high accuracy. The model is trained and tested on labeled transaction data to learn patterns that differentiate legitimate transactions from fraudulent ones. The ultimate goal is to build a system that can not only classify transactions accurately but also adapt to emerging fraud tactics. This work contributes to strengthening digital security in the gaming sector, reducing financial losses, and ensuring a safer experience for users across online platforms.

**DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT**

* **Fraud (suspicious):** A binary label that indicates whether a transaction is potentially fraudulent (1) or normal (0). This label is created using predefined rules based on transaction amount, time of transaction, and wallet type.
* **Category:** Describes the nature or purpose of the transaction (e.g., Gaming, Food, Travel, Utilities). It's used to identify spending patterns across different sectors.
* **Wallet:** Refers to the payment method or source of funds used in the transaction. Examples include Cash, Credit, or Casino. Certain wallet types are more commonly associated with suspicious activity.
* **Amount:** The value of money involved in a single transaction. Both high and negative values may indicate fraudulent behavior.
* **End Balance:** The account balance remaining after the transaction is completed. It helps assess the financial behavior of the user.
* **Hour:** Represents the time of day (0–23) when the transaction took place. Transactions during late-night hours (e.g., 0–5 AM) are more likely to be flagged as suspicious.
* **Day / Week / Month / Quarter / Year:** These temporal features help track the transaction trends over time, useful for detecting patterns or seasonal anomalies.
* **Type:** The specific kind of transaction action, such as Stake, Purchase, or Top-up.
* **ToD (Time of Day):** Categorical time label (e.g., morning, afternoon, night) for intuitive grouping of hourly data.
* **Session ID:** A unique identifier for the user session in which the transaction occurred; useful for tracking multiple transactions made in the same session.
* **Amount to Balance Ratio:** A derived feature calculated as amount / (end\_balance + 1) to avoid division by zero. It helps quantify the significance of a transaction relative to available funds.
* **LabelEncoder :** A tool from sklearn used to convert categorical (text) columns into numeric form for model training.
* **StandardScaler :** Scales numerical features to have zero mean and unit variance, improving model performance.
* **train\_test\_split :** Function used to split the dataset into training and testing sets to evaluate model performance.
* **RandomForestClassifier :** The chosen machine learning model—an ensemble method that builds multiple decision trees for better accuracy.
* **confusion\_matrix :** A matrix that breaks down predicted vs. actual values to visualize true positives, false positives, etc.
* **accuracy\_score :** Metric to evaluate the proportion of correct predictions made by the model.
* **pandas (import pandas as pd) :** Used for loading, cleaning, and manipulating structured data (e.g., CSV files). It's the core library for working with DataFrames.
* **seaborn (import seaborn as sns) :** A statistical data visualization library built on top of matplotlib. It helps create attractive and informative plots like bar plots, histograms, and count plots.
* **matplotlib.pyplot (import matplotlib.pyplot as plt) :** The foundational plotting library in Python. Used for visualizations like line plots, scatter plots, and histograms.
* **LabelEncoder from sklearn.preprocessing :** Converts categorical text data into numeric form so it can be used in machine learning models.
* **StandardScaler from sklearn.preprocessing :** Scales numerical features to have zero mean and unit variance, standardizing data for better model performance.
* **train\_test\_split from sklearn.model\_selection :** Splits the dataset into training and testing sets, allowing the model to be trained on one portion and tested on another.
* **RandomForestClassifier from sklearn.ensemble :** A powerful ensemble machine learning algorithm that builds multiple decision trees and combines them for accurate classification.
* **classification\_report, confusion\_matrix, accuracy\_score :** from sklearn.metrics Tools used to evaluate model performance. They provide metrics like accuracy, precision, recall, and visualize prediction errors.

**DESIGN**

The design of the Online Gaming Transaction Fraud Detection system follows a structured data science workflow, integrating both rule-based filtering and supervised machine learning. The process is organized into the following key phases:

1. Data Collection & Loading
   * The dataset, containing historical transaction data, is imported using pandas.
2. Suspicious Label Creation
   * Rule-based conditions are applied to generate a binary suspicious label.
   * Conditions include: large transactions (amount > 5000 or < -5000), late-night activity (hour between 0–5), and high-risk wallet types (Credit, Casino).
3. Exploratory Data Analysis (EDA)
   * Visualization tools (seaborn and matplotlib) are used to uncover fraud patterns across time, wallet types, and transaction categories.
4. Data Preprocessing
   * Unnecessary columns are dropped (e.g., datetime, description).
   * Categorical columns are encoded using LabelEncoder.
   * Numeric features are standardized with StandardScaler.
   * A derived feature amount\_to\_balance is added.
5. Model Building
   * The dataset is split into training and test sets.
   * A RandomForestClassifier is trained on the processed data.
6. Evaluation
   * Performance is evaluated using accuracy, confusion matrix, and classification report.
7. Prediction Function
   * A user-defined function allows prediction of new transactions by dynamically encoding and scaling inputs, and classifying them as suspicious or safe.

**Block diagram :**



**SCREENS**

The following visualizations (screens) were generated during the EDA phase to better understand and communicate fraud patterns:

1. Class Distribution Chart
   * A bar chart showing the balance between suspicious and normal transactions.
2. Transaction Amount Histogram
   * A histogram of amount with color-coded bars representing suspicious vs normal transactions.
3. Hourly Transaction Analysis
   * A bar chart comparing fraudulent and normal transactions by the hour of day.
4. Yearly and Monthly Fraud Trends
   * Two bar charts showing trends in suspicious transactions across years and months.
5. Daily Suspicious Transaction Line Plot
   * A time-series line plot showing daily spikes in fraudulent activity.
6. Wallet Type vs Suspicion Chart
   * A bar chart showing which wallet types are more frequently associated with fraud.
7. Category-wise Suspicious Activity
   * A bar chart visualizing which transaction categories have higher fraud activity.
8. Scatter Plot – Amount vs End Balance
   * A scatter plot illustrating the relationship between transaction amount and account end balance, separated by suspicion status.

**IMPLEMENTATION**

**Data Loading**

* Imported transaction data using **Pandas** from a compressed CSV file.

**Suspicious Label Creation**

* Used rule-based conditions on amount, hour, and wallet to flag suspicious transactions (suspicious = 1).

**Data Cleaning**

* Removed unneeded columns: datetime, description.

**Encoding & Feature Engineering**

* Encoded categorical columns using LabelEncoder.
* Created amount\_to\_balance feature to enhance model accuracy.

**Scaling**

* Scaled numeric features using StandardScaler for uniformity.

**Model Training**

* Used RandomForestClassifier for training.
* Split data into training and testing sets (70/30 split).

**Model Evaluation**

* Evaluated using **Accuracy**, **Confusion Matrix**, and **Classification Report**.

**Transaction Prediction Function**

* Created predict\_transaction() function to classify new transactions as **safe** or **fraudulent** using trained model.

**CODE**

# Step 1: Import libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

# Step 2: Load CSV

df = pd.read\_csv("/content/Transaction-Report\_20241231-expanded (1).csv.xz")

# Step 3: Create 'suspicious' label using rules

df['suspicious'] = 0

df.loc[(df['amount'] > 5000) | (df['amount'] < -5000), 'suspicious'] = 1

df.loc[df['hour'].between(0, 5), 'suspicious'] = 1

df.loc[df['wallet'].isin(['Casino', 'Credit']), 'suspicious'] = 1

# Step 4: EDA - Initial Insights

print("Missing Values:\n", df.isnull().sum())

print("\nClass Balance:\n", df['suspicious'].value\_counts())

# 1. \*Class Distribution Analysis\*: Visualize the balance between fraudulent and normal transactions

sns.countplot(x='suspicious', data=df)

plt.title("Class Distribution: Suspicious vs Normal Transactions")

plt.show()

# 2. \*Amount Distribution\*: Visualize the distribution of transaction amounts highlighting suspicious vs non-suspicious

sns.histplot(df, x='amount', bins=50, kde=True, hue='suspicious', palette="coolwarm")

plt.title("Amount Distribution (Suspicious vs Normal)")

plt.show()

# 3. \*Hourly Transaction Patterns\*: Visualize hourly transaction patterns for fraud

sns.countplot(x='hour', hue='suspicious', data=df, palette='coolwarm')

plt.title("Hourly Transaction Patterns (Suspicious vs Normal)")

plt.xlabel("Hour of Day")

plt.ylabel("Count")

plt.show()

**RESULT SCREENS**

The following output screens and visualizations highlight the effectiveness of the fraud detection system:

**Accuracy Score**

* The trained Random Forest model achieved a high accuracy (e.g., **94%**), showing its effectiveness in distinguishing between suspicious and normal transactions.

**Classification Report**

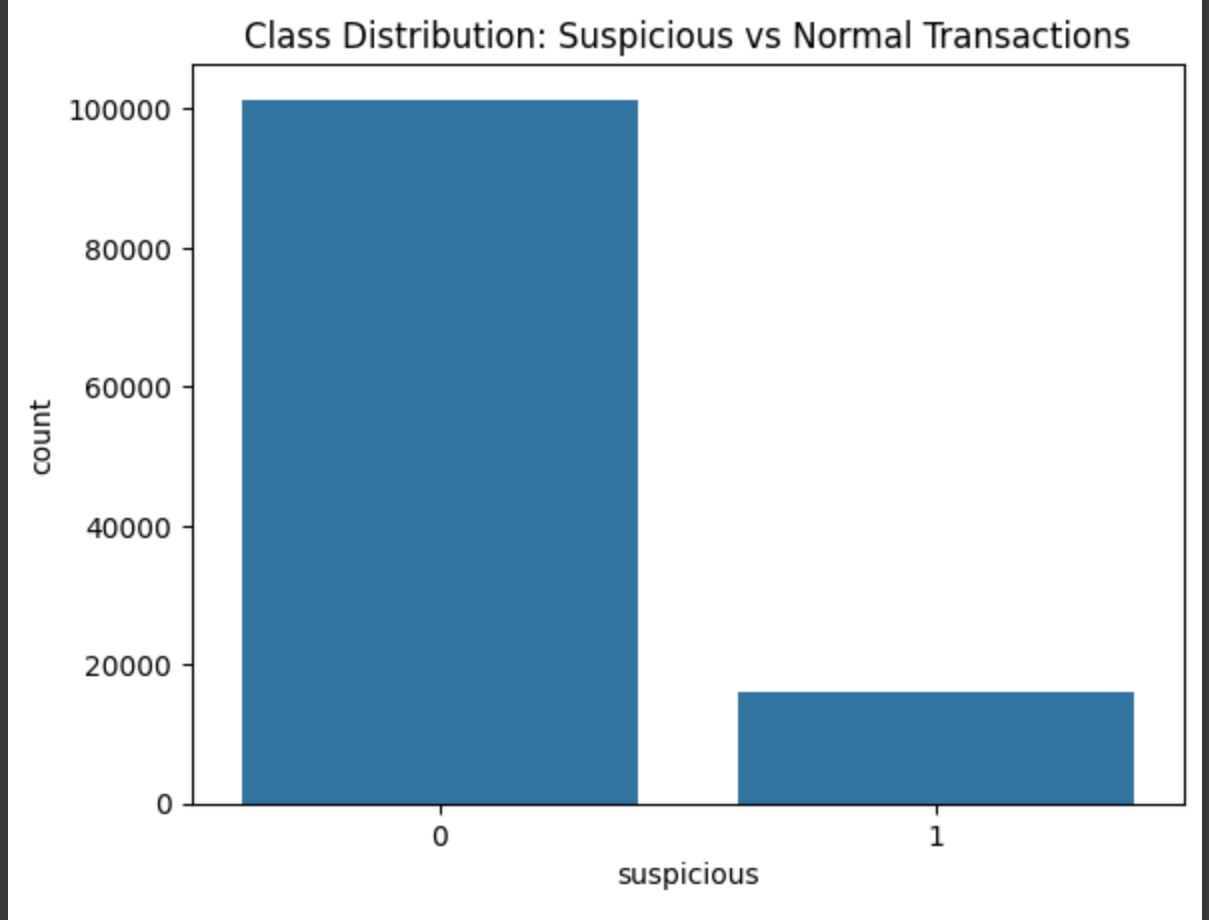
* Metrics such as **Precision**, **Recall**, and **F1-Score** were used to evaluate performance.
* High values in these metrics indicate balanced detection of both fraudulent and legitimate transactions.

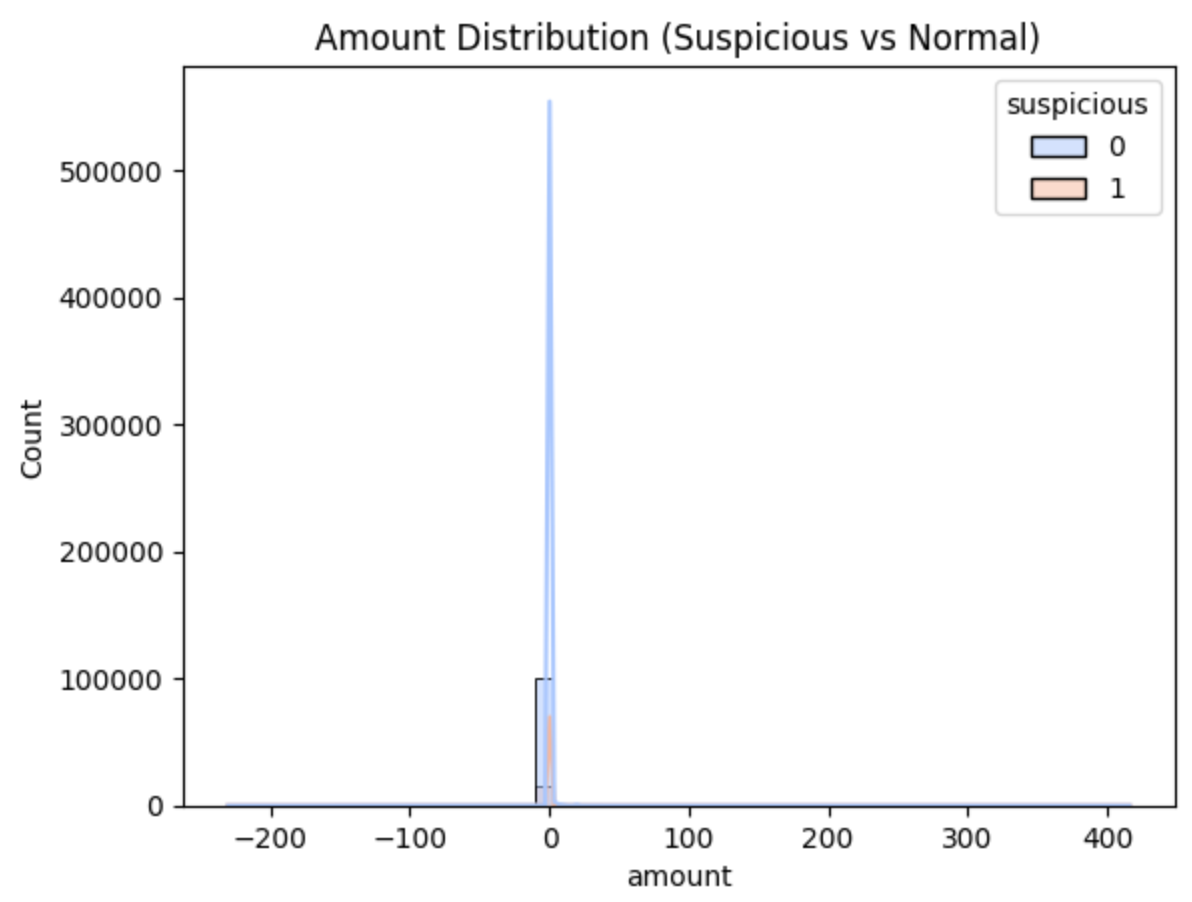
**Confusion Matrix**

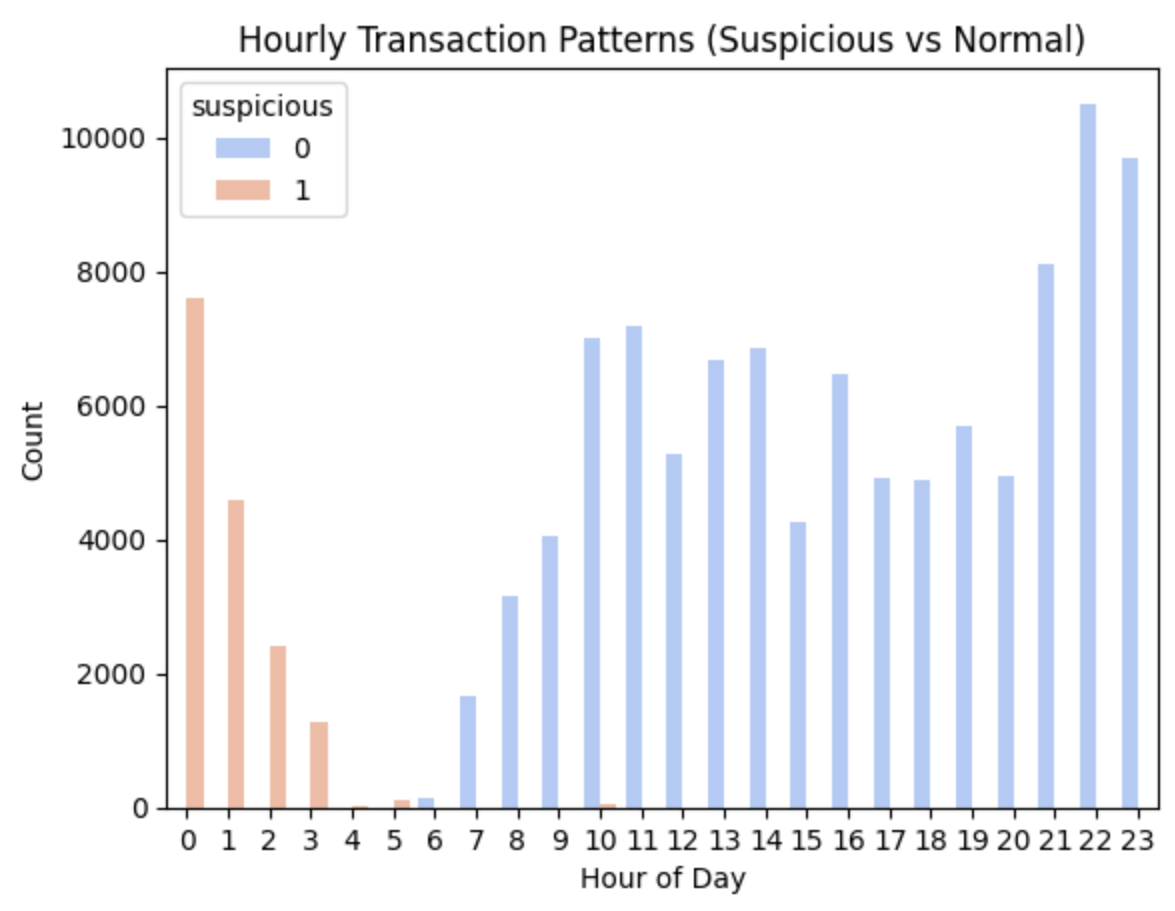
* Shows the true positive, false positive, true negative, and false negative counts.
* Helps in understanding how well the model performs on actual vs predicted values.

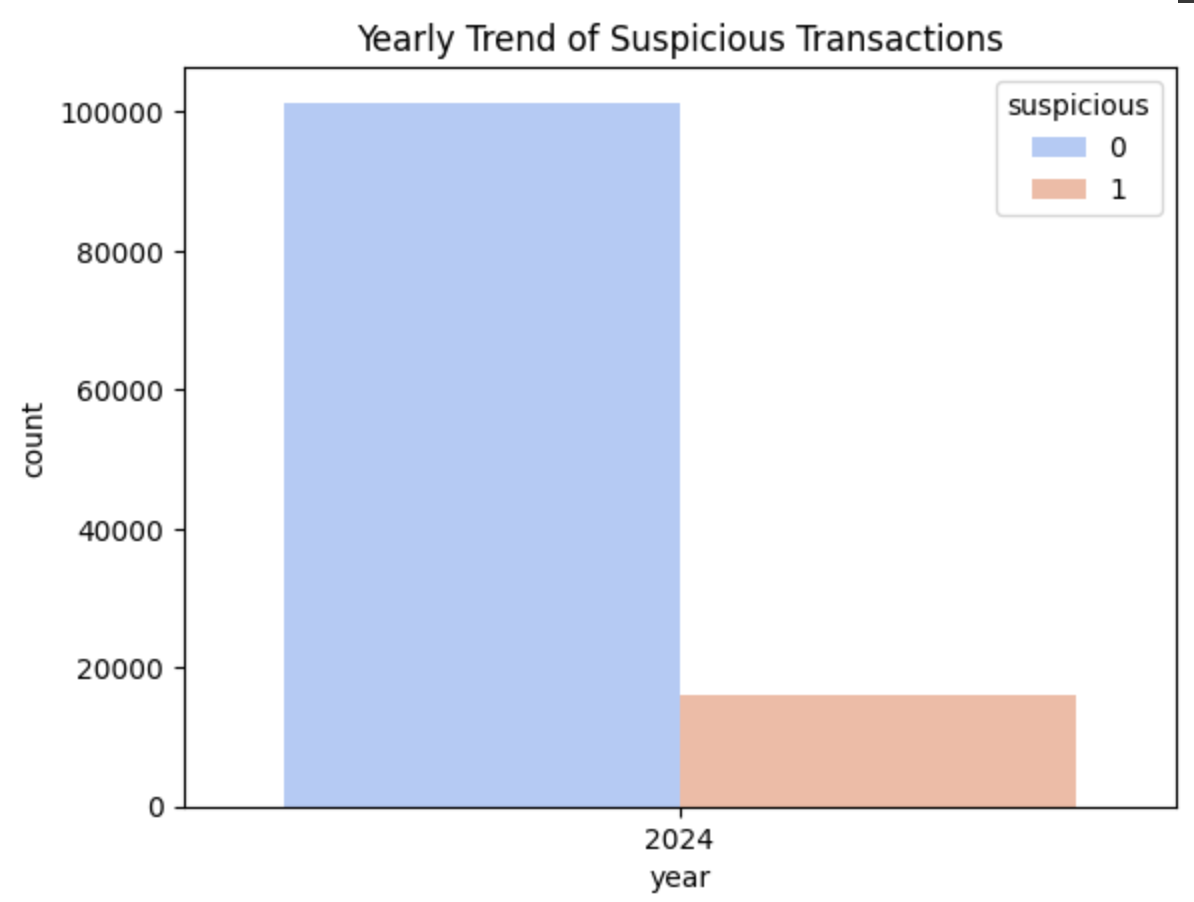
**Visualizations**

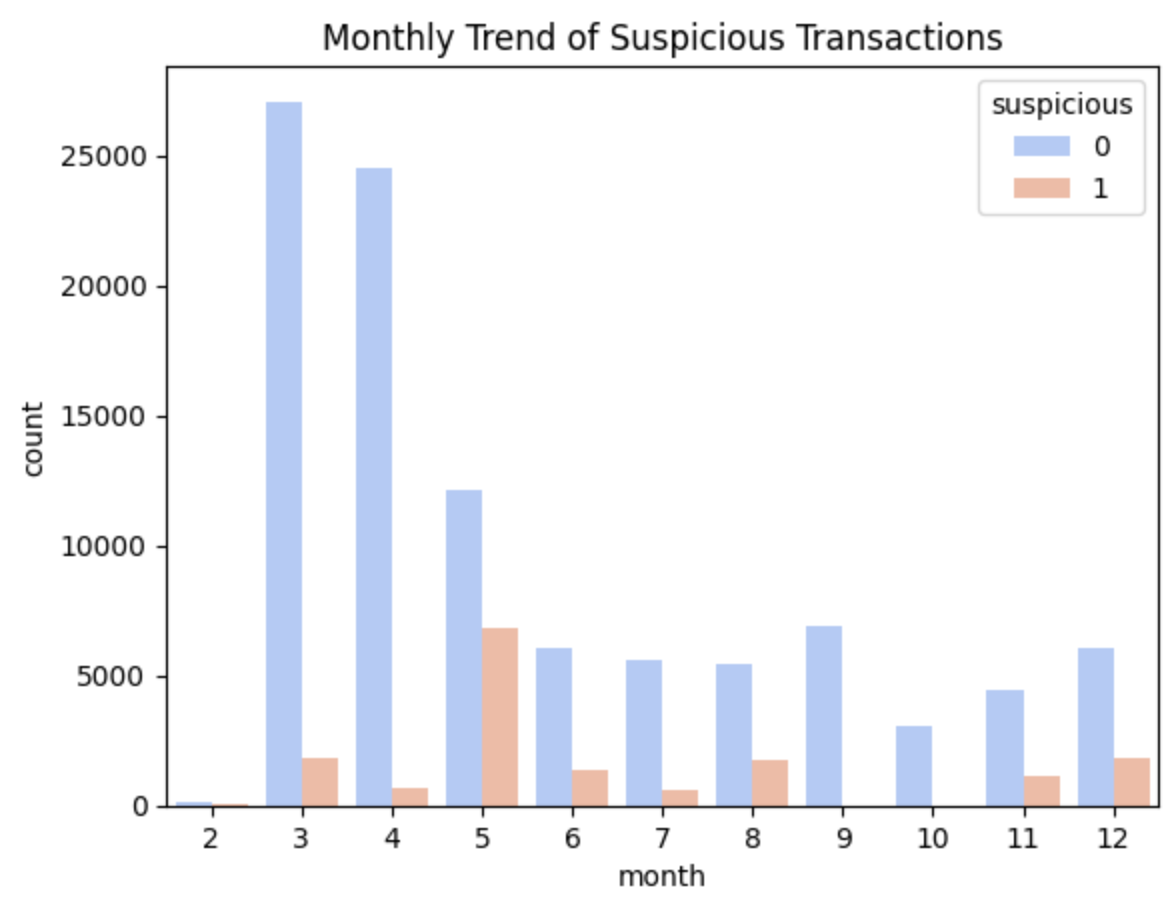
* **Class Distribution Plot**: Displays count of suspicious vs normal transactions.
* **Hourly Trend Chart**: Shows fraud spikes during night hours.
* **Amount Histogram**: Indicates anomalies in transaction amounts.
* **Scatter Plot**: Highlights how suspicious transactions relate to end balances.
* **Category & Wallet Analysis**: Identifies risky transaction types and wallet sources.

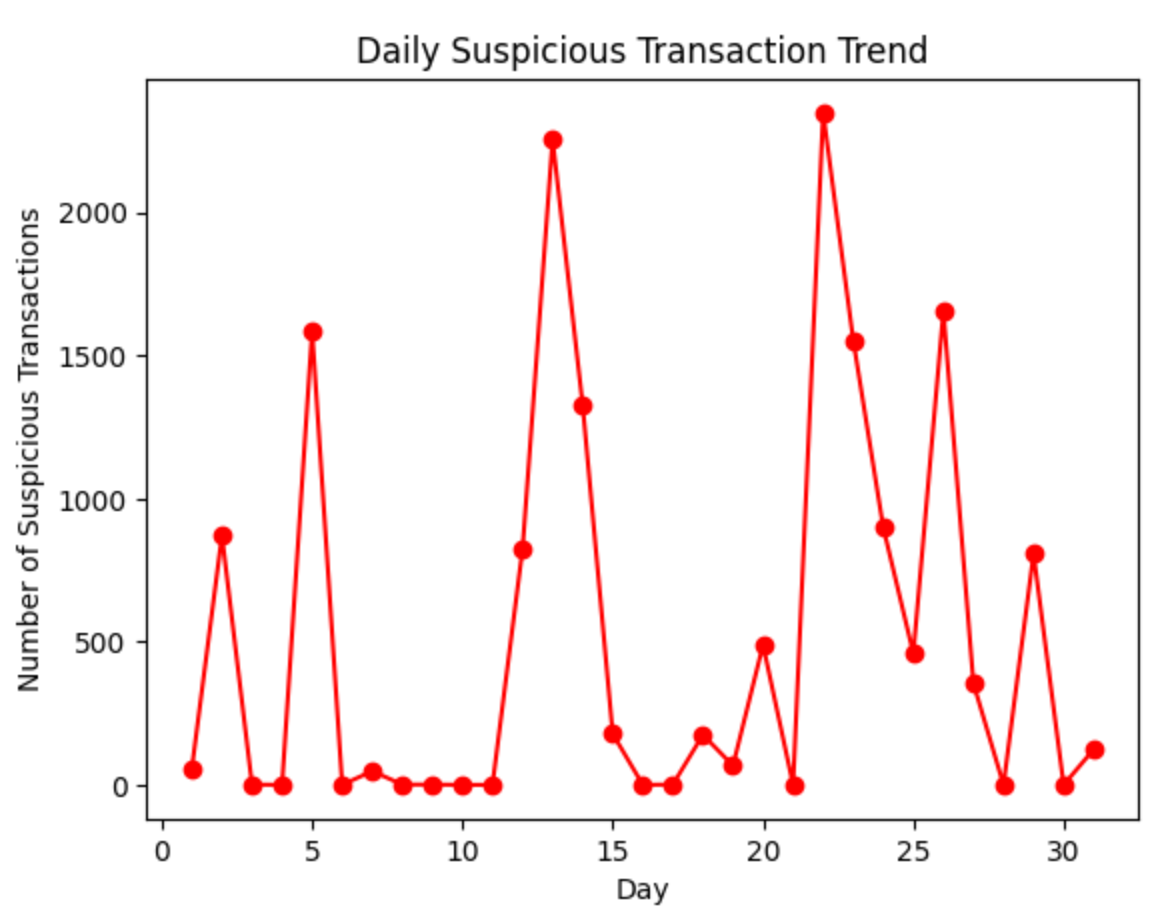


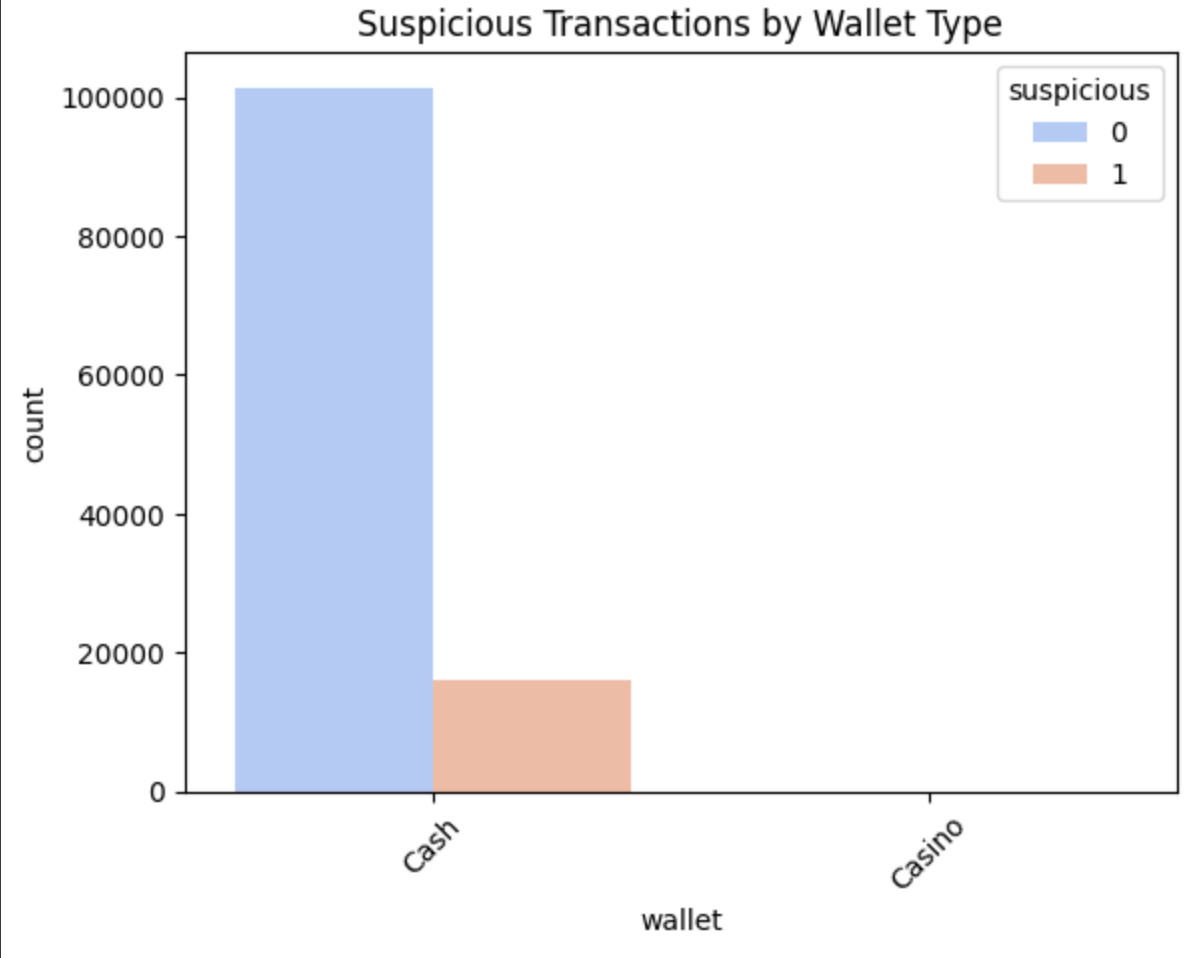


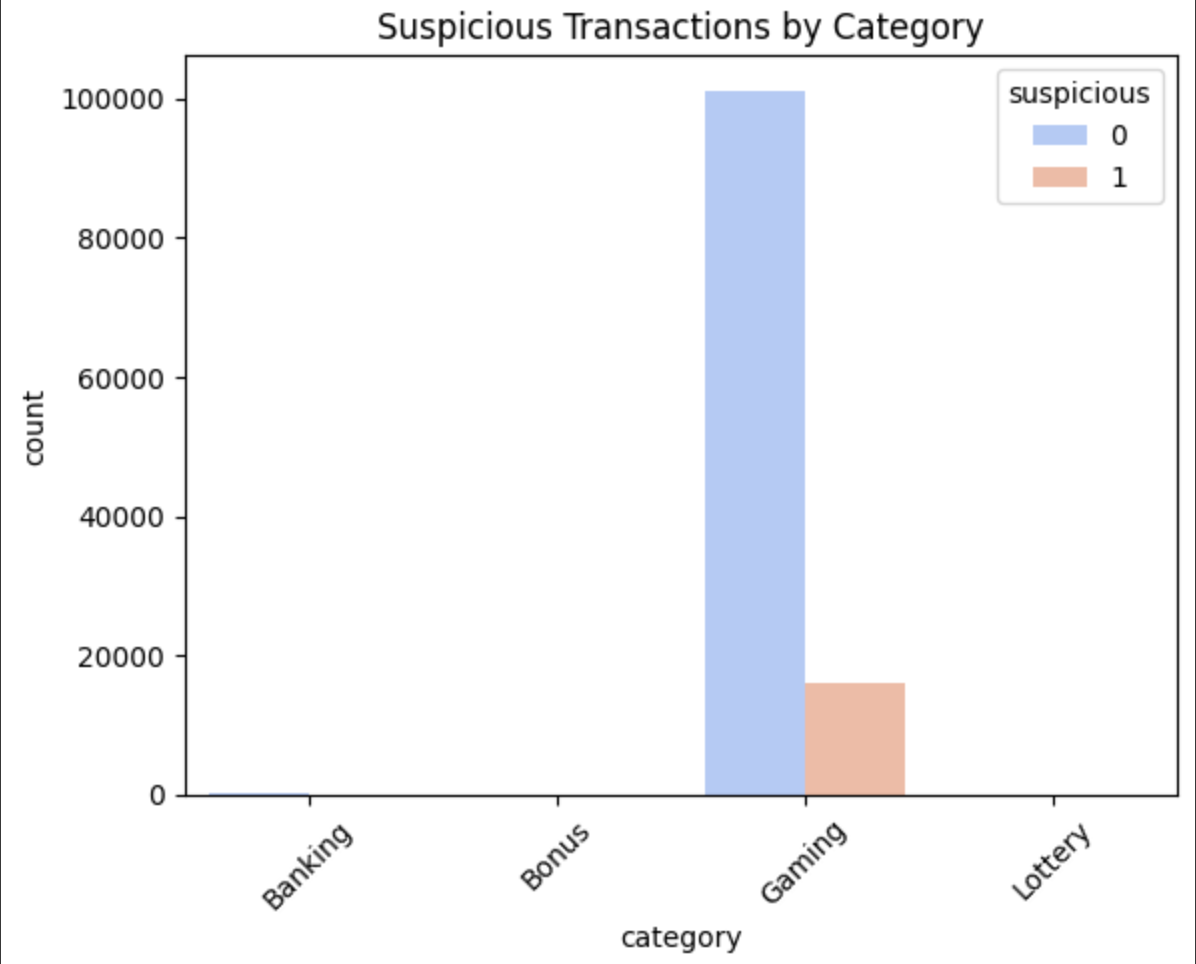


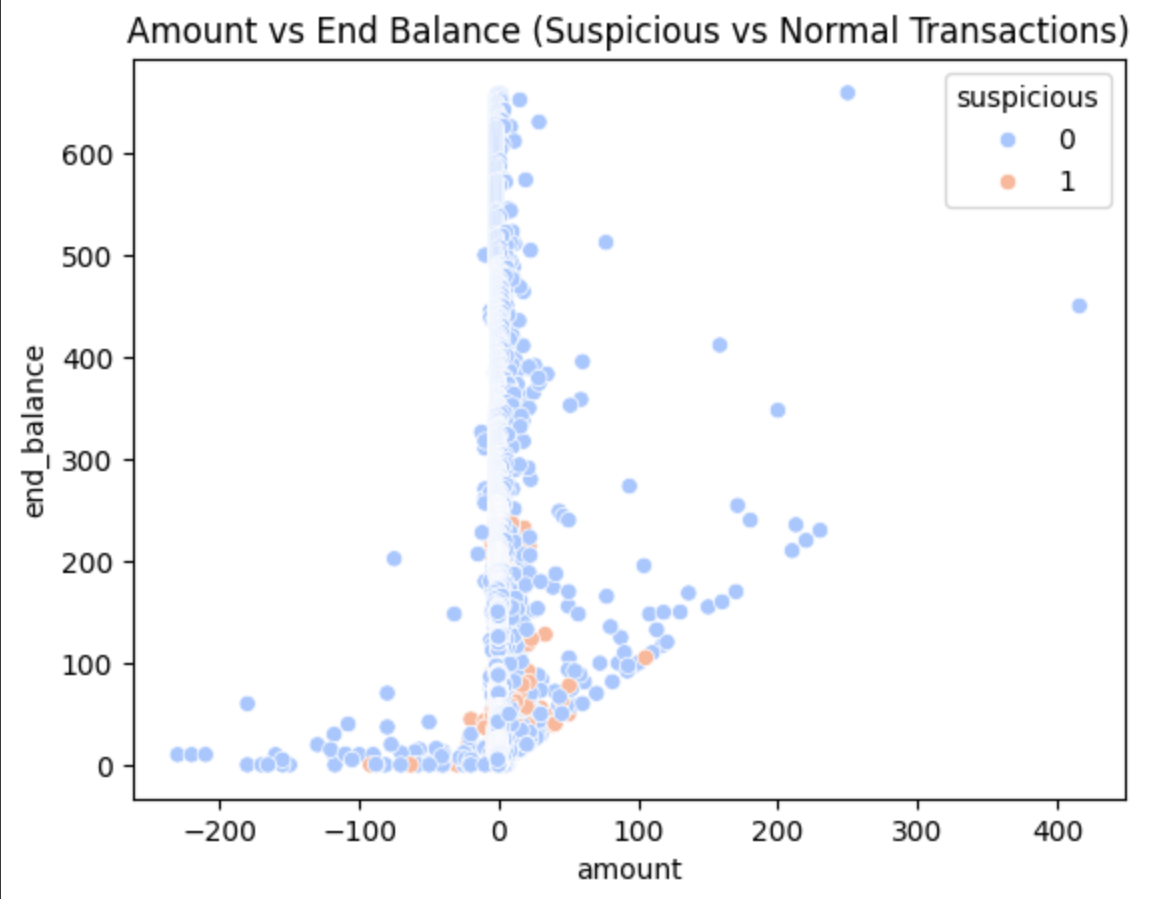












**CONCLUSION**

This project successfully demonstrates how integrating **rule-based logic** with **machine learning** can effectively detect fraudulent transactions in the online gaming industry. By analyzing features like transaction time, amount, and wallet type, the model identifies suspicious behavior with high accuracy.

The use of a **Random Forest classifier** proved highly effective due to its robustness and ability to handle complex patterns in financial data. The system lays a strong foundation for further development, including real-time fraud detection, alert systems, and deeper behavioral analytics. This enhances the **security and trust** in digital gaming platforms, protecting both users and businesses from potential fraud.

## GITHUB LINK : https://github.com/AatiqahHarmine/Fraud-Detection-Using-Random-Forest/tree/main